2. Xen installation

This section describes how to install Xen 4.1.1 from source. Alternatively, there may be pre-built packages available as part of your operating system distribution.

2.1. Preparation

In order to successfully build and run Xen we need to download and install some tools and utilities. To install them in Debian Squeeze we run the following command as root:

```
root@dom0$ apt-get install bcc bin86 gawk bridge-utils iproute libcurl3
libcurl4-openssl-dev bzip2 module-init-tools transfig tgif texinfo texlive-latex-
base texlive-latex-recommended texlive-fonts-extra texlive-fonts-recommended
pciutils-dev mercurial build-essential make gcc libc6-dev zlib1g-dev python
python-dev python-twisted libncurses5-dev patch libvncserver-dev libsdl-dev
libjpeg62-dev iasl libbz2-dev e2fslibs-dev git-core uuid-dev ocaml libx11-dev
bison flex xz-utils ocaml-findlib
```

If you have a 64 bit version of Debian/Ubuntu you also need this additional package:

```
root@dom0$ apt-get install gcc-multilib
```

To be able to use other operating systems than Linux/ FreeBSD as a guest (e.g. Windows) our hardware must has support for VT (Virtualization Technology). Hardware assisted virtualization offers new instructions to support direct calls by a paravirtualized guest/driver into the hypervisor, typically used for I/O or other hypercalls. The main idea behind this is to introduce a new privilege level, called the level -1 below the level 0. The VMM can run on this new level. By introducing this new level, the guest operating systems can run at level 0 without any modifications and the hardware requests they perform can be captured directly from the system. To check if our Intel based system supports this feature we can run:

```
root@dom0$ cat /proc/cpuinfo | grep vmx
```

For AMD based processors, we can run the following command:

```
root@dom0$ cat /proc/cpuinfo | grep svm
```

2.2. Installation

We are now ready to download and install Xen. In this guide we will use the version 4.1.1 of Xen. To get the source code we run:

```
user@dom0$ wget http://bits.xensource.com/oss-xen/release/4.1.1/xen-4.1.1.tar.gz
```

Now we can extract it and build it:

```
user@dom0$ tar xvf xen-4.1.1.tar.gz
user@dom0$ cd xen-4.1.1
user@dom0$ make xen
user@dom0$ make tools
user@dom0$ make stubdom
```

After the build has completed we should have a top-level directory called 'dist' in which all resulting targets will be placed. To install them on our system we run:

```
root@dom0$ make install-xen
root@dom0$ make install-tools PYTHON_PREFIX_ARG=
root@dom0$ make install-stubdom
```

At this point Xen and its utilities are installed on our system. We should have the following files in '/boot' directory:

```
/boot/xen.gz
/boot/xen-4.gz
/boot/xen-4.1.gz
/boot/xen-4.1.1.gz
```

Now, in the file '/etc/xen/xend-config.sxp' we enable the following option:

```
(xend-unix-server yes)
```

And finally, to enable automatic start of Xen related services on system startup we run:

```
root@dom0$ update-rc.d xencommons defaults 19 18
root@dom0$ update-rc.d xend defaults 20 21
root@dom0$ update-rc.d xendomains defaults 21 20
root@dom0$ update-rc.d xen-watchdog defaults 22 23
```

3. Creating and installing a dom0 kernel

At this point Xen is built and installed on our system but it is not ready for use, as we don't have a dom0 kernel yet. The kernel with which our system booted will not work with Xen. So, we are going to compile a dom0 kernel.

There are two different types of Xen dom0 capable kernels available today:

- **pvops** (paravirt_ops) kernels, featuring new rewritten Xen support based on the upstream (kernel.org) Linux pvops framework. This work has been included in upstream kernel.org kernel since Linux 2.6.37. Pvops is a piece of Linux kernel infrastructure to allow it to run paravirtualized on a hypervisor. This feature allows us to build a single Linux kernel binary which will either boot native on bare hardware, or boot fully paravirtualized as a Xen dom0 or domU.
- **xenlinux** kernels based on the "old" patches originally for Linux 2.6.18. These Xenlinux patches won't be integrated to upstream Linux.

In this guide we will use the latest stable Linux kernel which is 3.0.4 and has build-in support for pvops. The first thing to do is to grab the kernel from kernel.org:

```
user@dom0$ wget http://www.kernel.org/pub/linux/kernel/v3.0/linux-3.0.4.tar.gz
user@dom0$ tar xvf linux-3.0.4.tar.gz
```

Now, we can configure our new kernel by running:

```
user@dom0$ make menuconfig
```

We must be sure that we have a correct Processor family set in the kernel configuration. Xen dom0 options won't show up at all if we have too old CPU selected. If we are building a 32 bit version of the kernel we need to enable PAE support:

```
Processor type and features →
        High memory support (64GB)
        PAE (Physical Address Extension) Support - enabled
```

Note that PAE is not needed for 64 bit kernels. Also, if we are building a 32 bit kernel we need to set the option 'CONFIG_HIGHPTE=n':

```
Processor type and features →
        Allocate 2nd-level pagetables from highmem - disabled
```

Furthermore, Xen Dom0 support depends on ACPI support on both 32 and 64 bits versions of the Linux kernel. Thus, we must enable ACPI support during configuration:

```
ACPI (Advanced Configuration and Power Interface) Support -
enabled
```

Below, is a list of options needed to compile Linux kernel with dom0 support:

- CONFIG_ACPI_PROCFS=y
- CONFIG_XEN=y
- CONFIG_XEN_MAX_DOMAIN_MEMORY=32
- CONFIG_XEN_SAVE_RESTORE=y
- CONFIG_XEN_DOM0=y
- CONFIG_XEN_PRIVILEGED_GUEST=y
- CONFIG_XEN_PCI=y
- CONFIG_PCI_XEN=y
- CONFIG_XEN_BLKDEV_FRONTEND=y
- CONFIG_XEN_NETDEV_FRONTEND=y
- CONFIG_XEN_KBDDEV_FRONTEND=y
- CONFIG_HVC_XEN=y
- CONFIG_XEN_FBDEV_FRONTEND=y
- CONFIG_XEN_BALLOON=y
- CONFIG_XEN_SCRUB_PAGES=y
- CONFIG_XEN_DEV_EVTCHN=y
- CONFIG_XEN_GNTDEV=y
- CONFIG_XEN_BACKEND=y
- CONFIG_XEN_BLKDEV_BACKEND=y
- CONFIG_XEN_NETDEV_BACKEND=y
- CONFIG_XENFS=y
- CONFIG_XEN_COMPAT_XENFS=y
- CONFIG_XEN_XENBUS_FRONTEND=y
- CONFIG_XEN_PCIDEV_FRONTEND=y

To enable these options we can select the following additional fields during configuration:

```
Processor type and features →
        Paravirtualized guest support [y] →
                Xen guest support – enabled

Bus oprions (PCI etc.)→
```

```
        Xen PCI frontend – enabled

Device Drivers →
        Block Devices [*] →
                Xen virtual block device support – enabled
                Block-device backend driver – enabled
        Network device support [*] →
                Xen network device frontend driver – enabled
                Xen backend network device – enabled
        Input device support →
                Miscellaneous devices →
                        Xen virtual keyboard and mouse support –
                        enabled
        Character devices →
                Xen Hypervisor console support – enabled

        Xen driver support →
                Xen memory balloon driver – enabled
                Scrub pages before returning them to system –
                enabled
                Xen /dev/xen/evtchn device Backend driver support
                – enabled
                Xen filesystem – enabled
                Create compatibility mount point /proc/xen –
                enabled
                Create xen entries under /sys/hypervisor – enabled
                userspace grant access device driver – enabled
                User-space grant reference allocator driver – enabled
                xen platform pci device driver – enabled
```

Now, we can build and install the kernel:

```
user@dom0$ make
root@dom0$ make modules_install
root@dom0$ make install
root@dom0$ cd /boot
root@dom0$ mkinitramfs -o initrd.img-3.0.4 3.0.4
root@dom0$ update-grub
```

We can now reboot our system to the Xen enabled 3.04 Linux kernel. To verify that Xen is running we can do the following:

```
root@dom0$ cat /proc/xen/capabilities
```

The output must be same as the following line:

```
control_d
```

This tells us that we have booted in a Xen control domain (dom0). Also, to verify that our Xen environment is working properly we can run:

```
root@dom0$ xm info
root@dom0$ xm list
```

If the output of these commands is the expected we have successfully installed Xen to our system. For example, 'xm list' command should list our dom0 domain:

```
Name                ID   Mem   VCPUs   State   Time(s)
Domain-0            0    1895  2       r-----  419.0
```